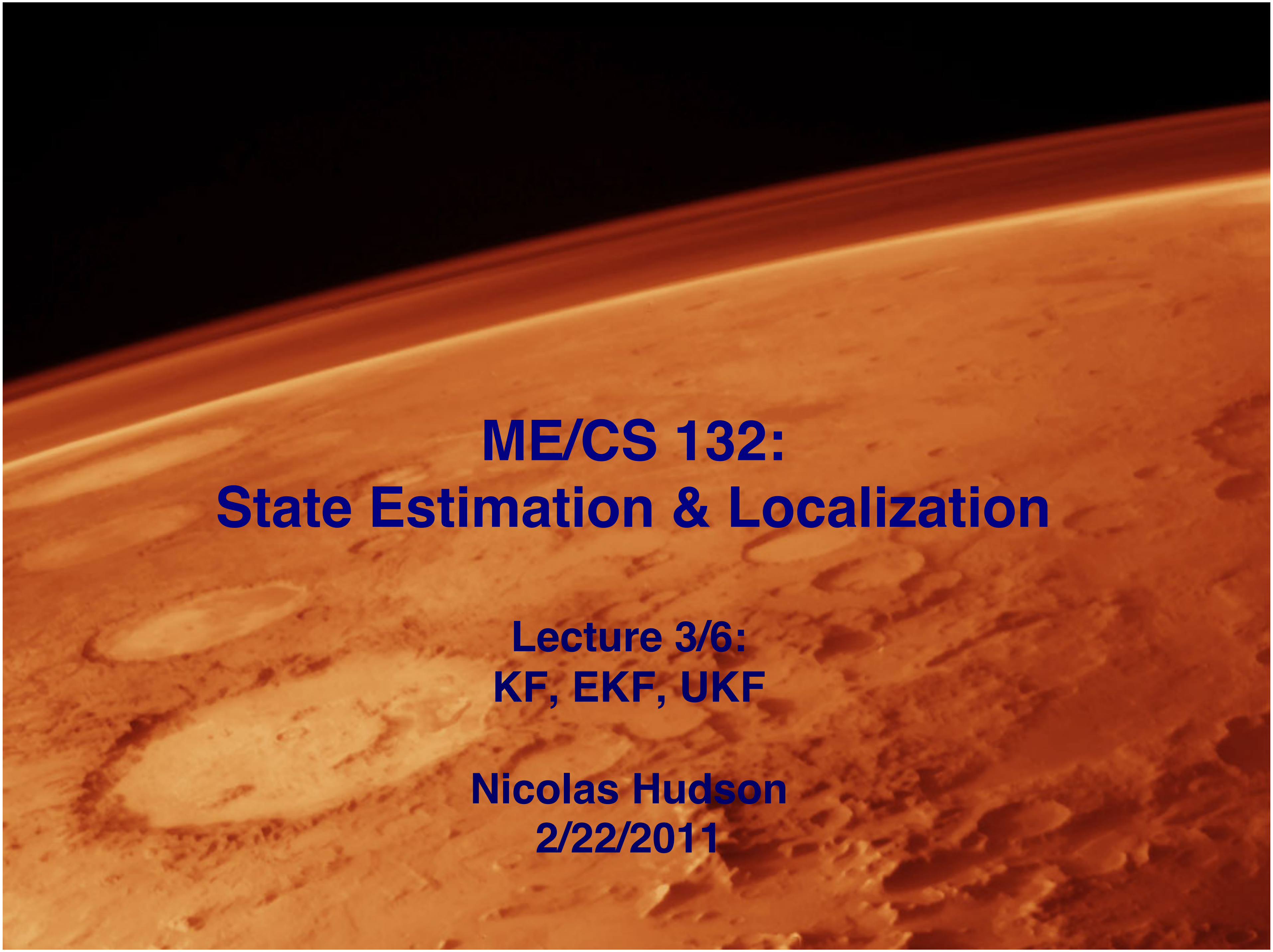


ME/CS 132: State Estimation & Localization

Lecture 3/6: KF, EKF, UKF

Nicolas Hudson

2/22/2011

TexPoint fonts used in EMF.

State Estimation & Localization Overview

- (1/6) Introduction
- (2/6) Linear Kalman Filter

- (3/6) TODAY: Extended Kalman Filter & Unscented KF
  - Review of Recursive Updating
  - Finish KF Proof
  - Non-Linear Systems
  - The EKF
  - The UKF

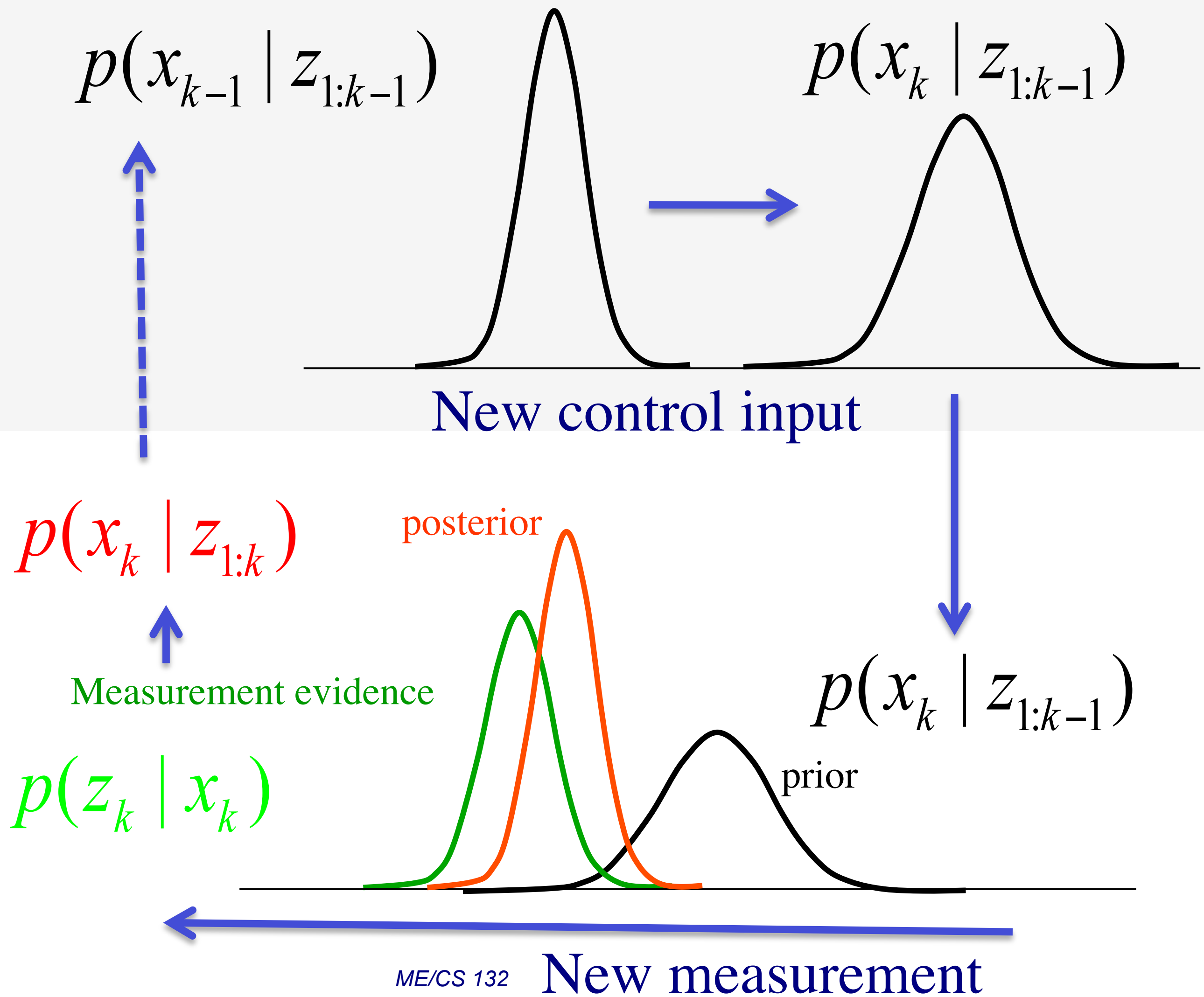
- (4/6) Particle Filters
- (5/6) Simultaneous Localization and Mapping (SLAM)
- (6/6) Issues in SLAM

Bayesian Recursion

Computation is done through a recursion:

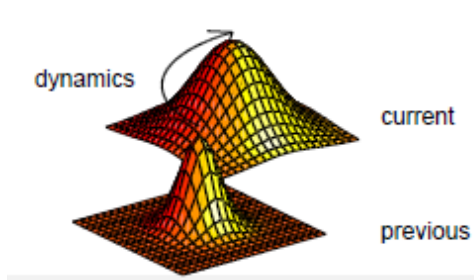
Prediction Step:

Update Step:



REVIEW: Optimal Bayesian Estimator:

Initialization: $p(x_0)$



Prediction:

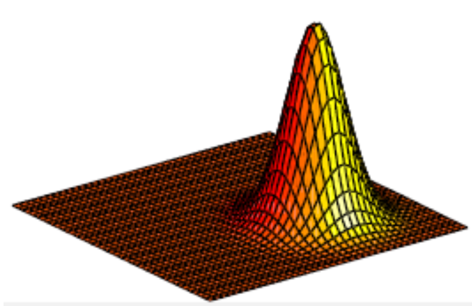
$$\boxed{p(x_k \mid z_{1:k-1}) = \int p(x_k \mid x_{k-1}) p(x_{k-1} \mid z_{1:k-1}) dx_{k-1}}$$

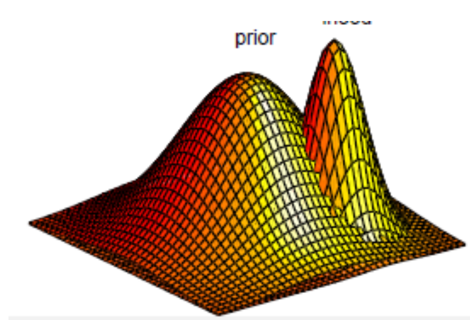
$k = k + 1$

z_k

Update:

$$\boxed{p(x_k \mid z_{1:k}) \propto p(z_k \mid x_k) p(x_k \mid z_{1:k-1})}$$





REVIEW KF: State Space Systems

Linear State Space Model (Gauss-Markov System)

Transition Function

$$x_k = A_k x_{k-1} + B_k u_k + q_k$$

$$z_k = C_k x_k + r_k$$

Measurement Model

Process Noise

$$q_k \sim N(0, Q)$$

$$r_k \sim N(0, R)$$

Measurement Noise

In probabilistic terms the model is

$$p(x_k \mid x_{k-1}) = N(A_k x_{k-1} + B_k u_k, Q)$$

$$p(z_k \mid x_k) = N(C_k x_k, R)$$

REVIEW: Kalman Filter: Prediction Step

prediction step:

$$\boxed{p(x_k \mid z_{1:k-1}) = \int p(x_k \mid x_{k-1})\, p(x_{k-1} \mid z_{1:k-1})\, dx_{k-1}}$$

$$N\left(A_k x_{k-1} + B_k u_k, Q_k\right) \quad N\left(\mu_{k-1}, \Sigma_{k-1}\right)$$

Posterior is just another Gaussian!
(Proof: will show later)

....

$$\boxed{p(x \mid z_{1:k-1}) = N\left(A_k \mu_{k-1} + B_k u_k \,,\, A_k \Sigma_{k-1} A_k^T + Q_k\right)}$$

REVIEW: Kalman Filter: Update Step

update step:

$$\boxed{p(x_k \mid z_{1:k}) = \frac{1}{\eta} p(z_k \mid x_k) p(x_k \mid z_{1:k-1})}$$

$$N(C_k x_k, R) \qquad\qquad N\left(\bar{\mu}_k, \bar{\Sigma}_k\right)$$

The product of two Gaussians -> is another Gaussian

$$p(x_k \mid z_{1:k}) = N(\mu_k, \Sigma_k)$$

(Proof: will show later)

$$\begin{cases} \mu_k = \bar{\mu}_k + K_k (z_k - C_k \bar{\mu}_k) \\ \Sigma_k = (I - K_k C_k) \bar{\Sigma}_k \end{cases} \quad \text{with} \quad K_k = \bar{\Sigma}_k C_k^T (C_k \bar{\Sigma}_k C_k^T + Q_k)^{-1}$$

REVIEW: Putting it all together : KF

Prior estimate $$p(x_{k-1} \mid z_{1:k-1}) = N(\mu_{k-1}, \Sigma_{k-1})$$

Prediction step
$$\bar{\mu}_k = A_k \mu_{k-1} + B_k u_k$$
$$\bar{\Sigma}_k = A_k \Sigma_{k-1} A_k^T + Q_k$$

Predicted estimate $$p(x_k \mid z_{1:k-1}) = N(\bar{\mu}_k, \bar{\Sigma}_k)$$

Update Step
$$K_k = \bar{\Sigma}_k C_k^T (C_k \bar{\Sigma}_k C_k^T + R_k)^{-1}$$
$$\mu_k = \bar{\mu}_k + K_k (z_k - C_k \bar{\mu}_k)$$
$$\Sigma_k = (I - K_k C_k) \bar{\Sigma}_k$$

Posterior estimate $$p(x_k \mid z_{1:k}) = N(\mu_k, \Sigma_k)$$

The EKF

- We will now look at the "**Extended Kalman Filter**"

- **You should read "Probabilistic Robotics" SS 3.3**

- The concept and derivations follows from the KF we just looked at.

Non-linear Systems:

Linear State Space Model

Transition Function

$$x_k = A_k x_{k-1} + B_k u_k + q_k$$

$$z_k = C_k x_k + r_k$$

Measurement Model

Process Noise

$$q_k \sim N(0, Q)$$

$$r_k \sim N(0, R)$$

Measurement Noise

Non-Linear State Space Model

Transition Function

$$x_k = g(x_{k-1}, u_k) + q_k$$

$$z_k = h(x_k) + r_k$$

Measurement Model

Process Noise

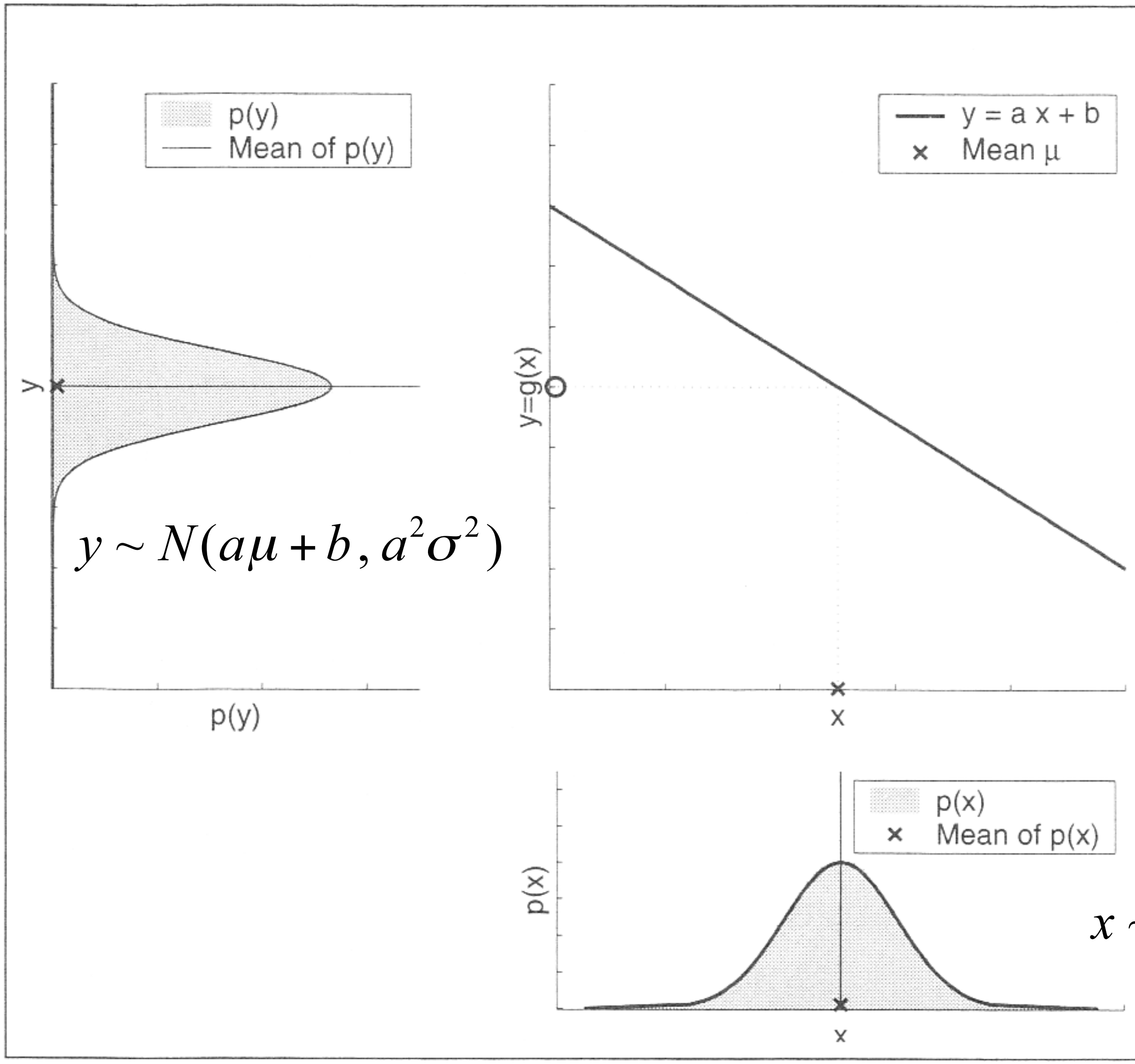
$$q_k \sim N(0, Q)$$

$$r_k \sim N(0, R)$$

Measurement Noise

Linear Function

$$y = g(x)$$
$$= ax + b$$

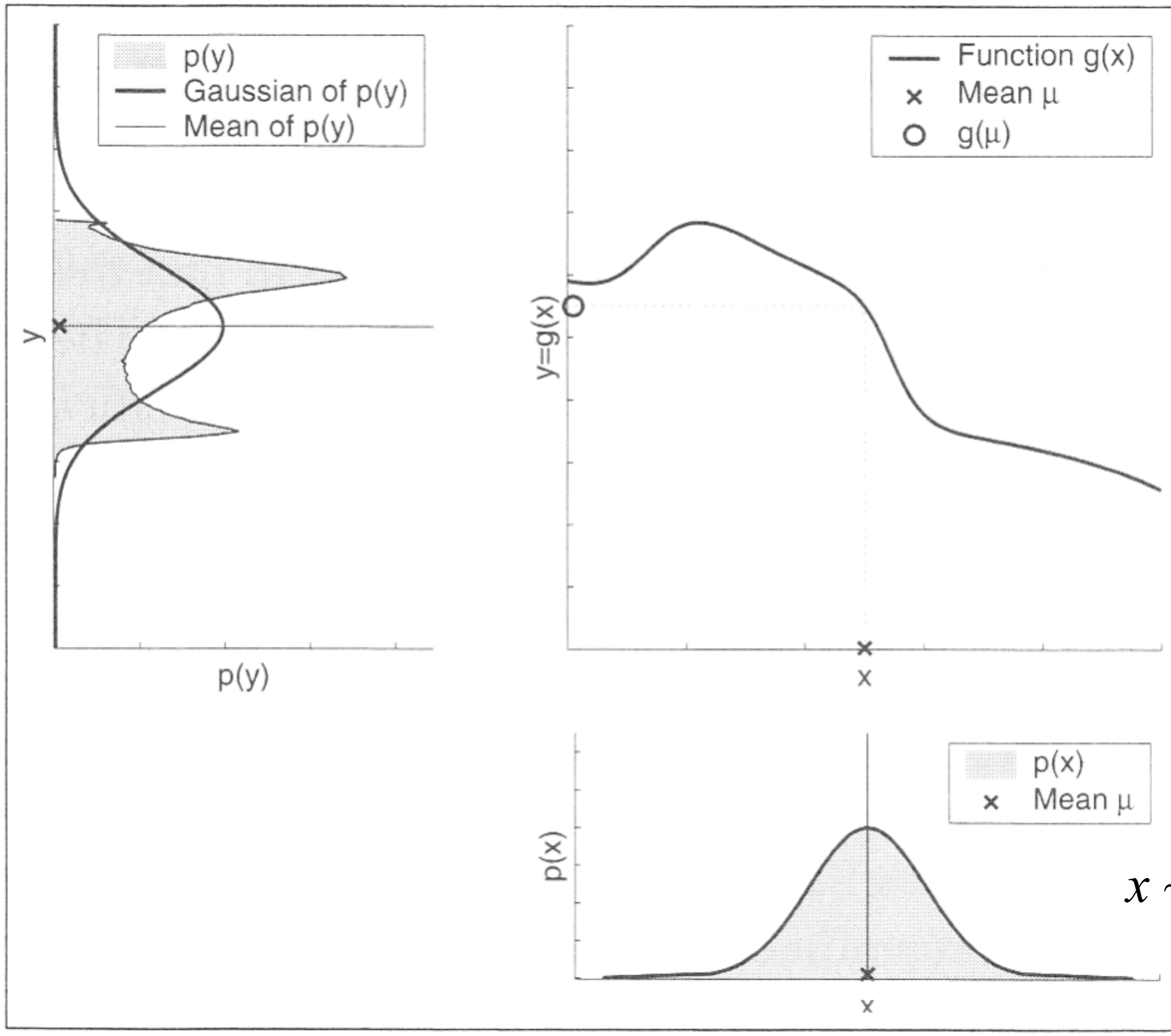


$$y \sim N(a\mu + b, a^2\sigma^2)$$

$$x \sim N(\mu, \sigma^2)$$

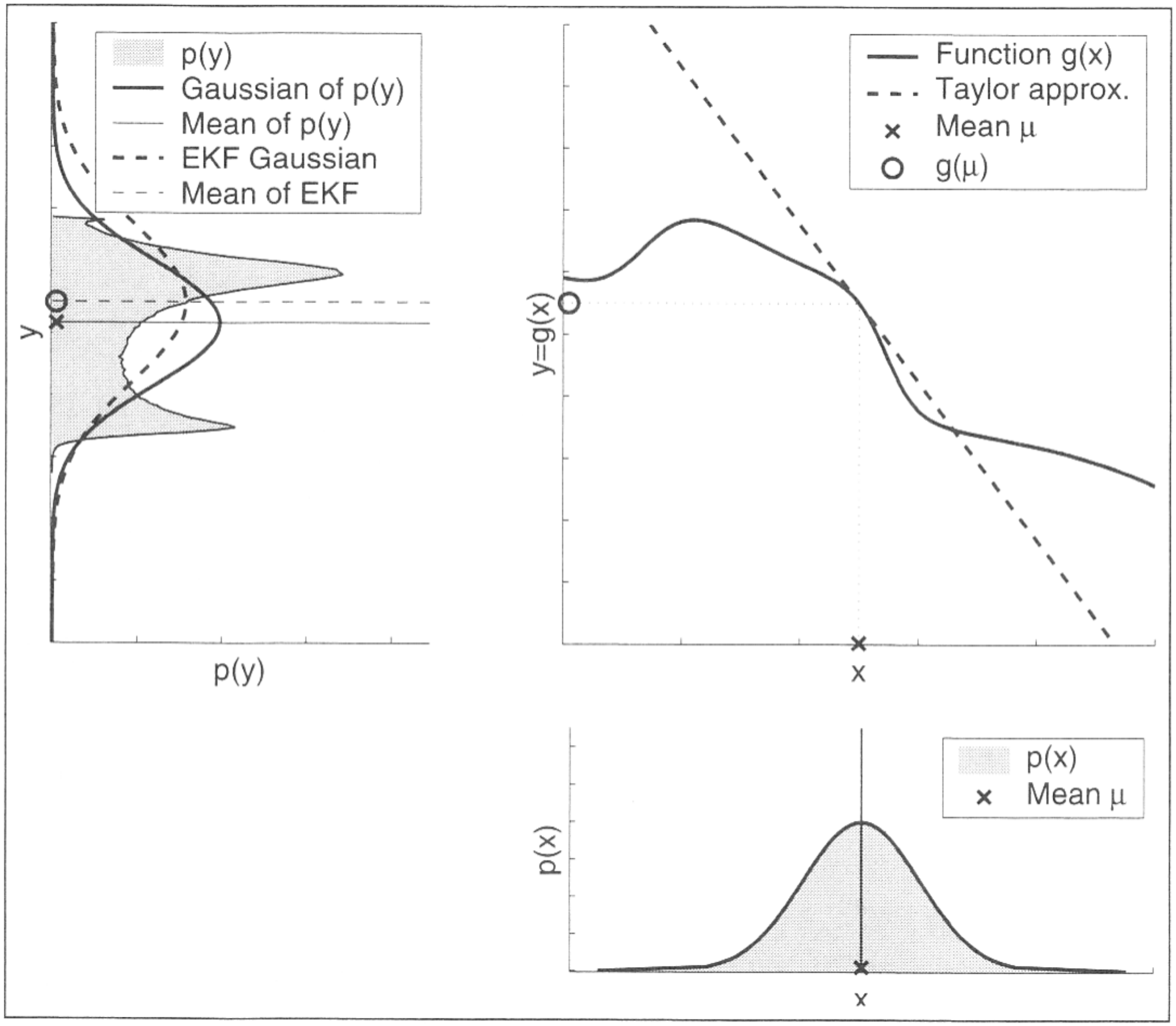
Non-Linear Function

$$y = g(x)$$



$$x \sim N(\mu, \sigma^2)$$

Linearization of Function



Linearization

- Relax the linear requirement of a Kalman Filter, by approximating the Propagation & Measurement functions with linear functions.

- We can linearize a function via a Taylor Expansion

Taylor Expansion

$$x_k = g(x_{k-1}, u_k) + q_k$$

$$z_k = h(x_k) + r_k$$

For a Gaussian prior, $x_{k-1} \sim N(\mu_{k-1}, \Sigma_{k-1})$
we will linearize about the most likely value μ_{k-1}

$$g(x_{k-1}, u_k) \approx g(\mu_{k-1}, u_k) + G_k(x_{k-1} - \mu_{k-1})$$

$$G_k \triangleq \left.\frac{\partial}{\partial x_{k-1}}\right|_{\mu_{k-1}} g(x_{k-1}, u_k)$$

Taylor Expansion

$$x_k = g(x_{k-1}, u_k) + q_k$$
$$z_k = h(x_k) + r_k$$

For a Gaussian predicted state, $\bar{x}_k \sim N(\bar{\mu}_k, \bar{\Sigma}_k)$
we will linearize about the most likely value $\bar{\mu}_k$

$$h(x_k) \approx h(\bar{\mu}_k) + H_k (x_k - \bar{\mu}_k)$$

$$H_k \triangleq \left. \frac{\partial}{\partial x_{k-1}} \right|_{\bar{\mu}_k} h(x_k)$$

Prediction and Update Formulas

prediction step:

$$\boxed{p(x_k \mid z_{1:k-1}) = \int p(x_k \mid x_{k-1})\, p(x_{k-1} \mid z_{1:k-1})\, dx_{k-1}}$$

$$N\left(g(\mu_{k-1}, u_k) - G_k(x_{k-1} - \mu_{k-1}), Q_k\right) \quad N\left(\mu_{k-1}, \Sigma_{k-1}\right)$$

Based on our knowledge of how to derive a Kalman filter, the answer is:

$$\boxed{p(x \mid z_{1:k-1}) = N\left(g(x_{k-1}, u_k), G_k \Sigma_{k-1} G_k^T + Q_k\right)}$$

....

(verify above Gaussian multiplication gives a quadratic form in $X = x_k, x_{k-1}$, and can use marginal properties of joint Gaussian distribution)

Prediction and Update Formulas

update step:

$$\boxed{p(x_k \mid z_{1:k}) = \frac{1}{\eta} p(z_k \mid x_k) p(x_k \mid z_{1:k-1})}$$

$$N\left(h(\bar{\mu}_k) - H_k(x_k - \bar{\mu}_k), R\right) \qquad N\left(\bar{\mu}_k, \bar{\Sigma}_k\right)$$

The product of two Gaussians -> is another Gaussian

$$p(x_k \mid z_{1:k}) = N(\mu_k, \Sigma_k)$$

(Proof: analogous to KF!- look at quadratic forms)

$$\begin{cases} \mu_k = \bar{\mu}_k + K_k(z_k - h(\bar{\mu}_k)) \\ \Sigma_k = (I - K_k H_k)\bar{\Sigma}_k \end{cases} \quad \text{with} \quad K_k = \bar{\Sigma}_k H_k^T (H_k \bar{\Sigma}_k H_k^T + Q_k)^{-1}$$

Putting it all together : EKF

Prior estimate $p(x_{k-1} \mid z_{1:k-1}) = N(\mu_{k-1}, \Sigma_{k-1})$

Prediction step

$$\bar{\mu}_k = g(\mu_{k-1}, u_k)$$
$$\bar{\Sigma}_k = G_k \Sigma_{k-1} G_k^T + Q_k$$

Predicted estimate $p(x_k \mid z_{1:k-1}) = N(\bar{\mu}_k, \bar{\Sigma}_k)$

Update Step

$$K_k = \bar{\Sigma}_k H_k^T (H_k \bar{\Sigma}_k H_k^T + R_k)^{-1}$$
$$\mu_k = \bar{\mu}_k + K_k (z_k - h(\bar{\mu}_k))$$
$$\Sigma_k = (I - K_k H_k) \bar{\Sigma}_k$$

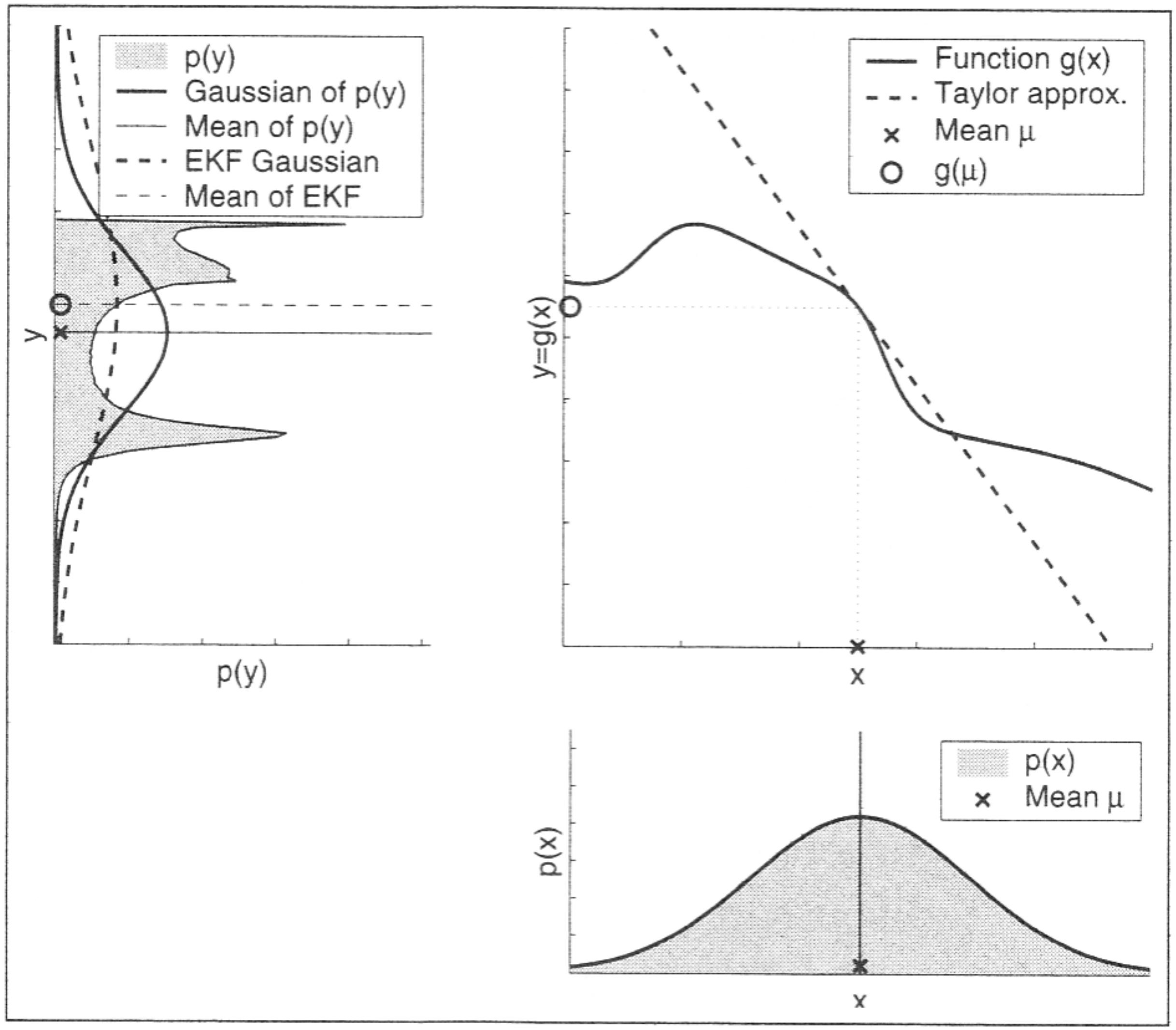
Posterior estimate $p(x_k \mid z_{1:k}) = N(\mu_k, \Sigma_k)$

EKF notes

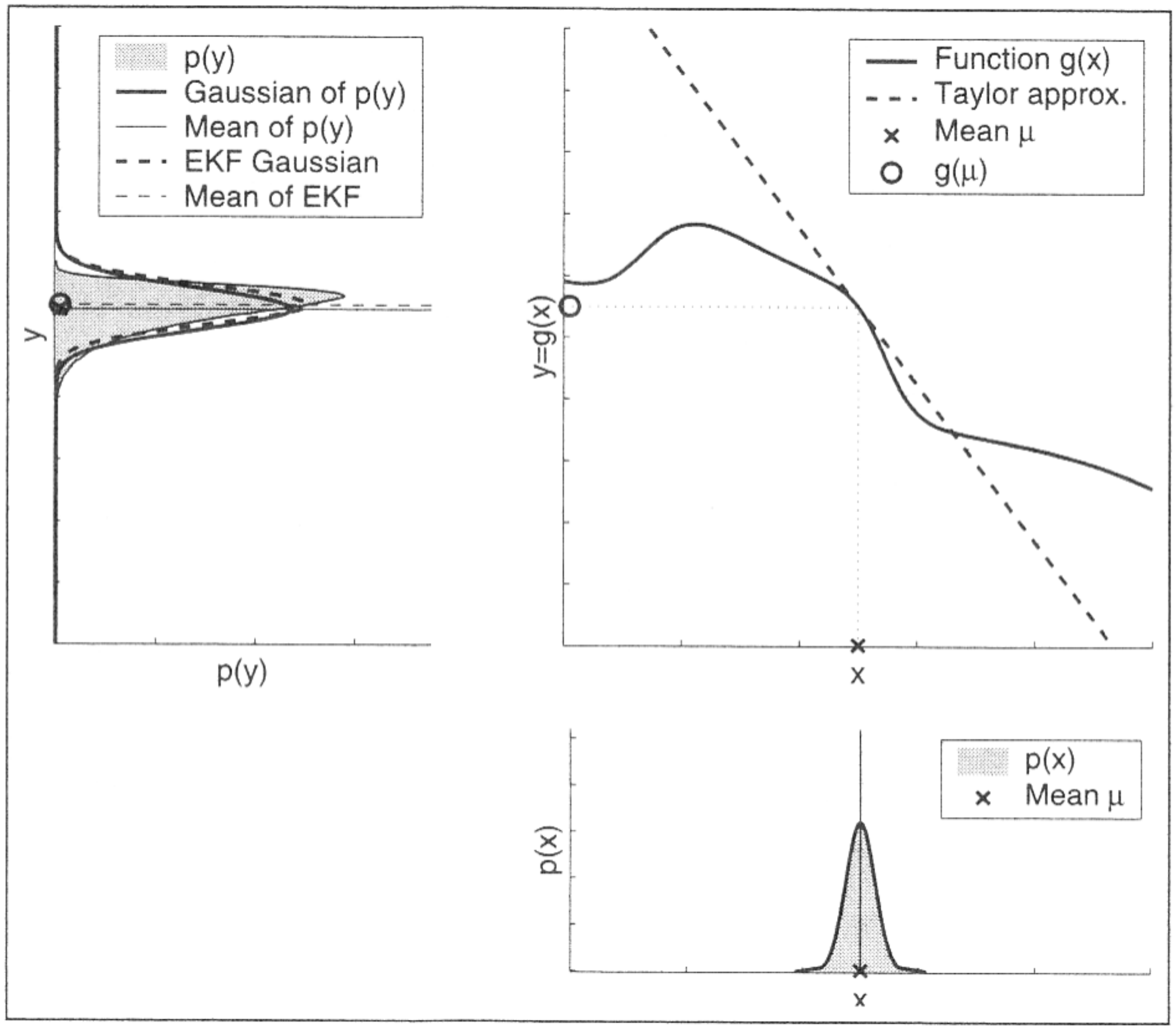
- The EKF is efficient as it uses only a Gaussian distribution for state representation.
- Very widely used - Almost all robotics problems are non-linear.

- The degree to which a EKF will work depends on how well the system linearize: depends on both the system uncertainty and the governing equations.

Linearization of Function (BAD)



Linearization of Function (GOOD)

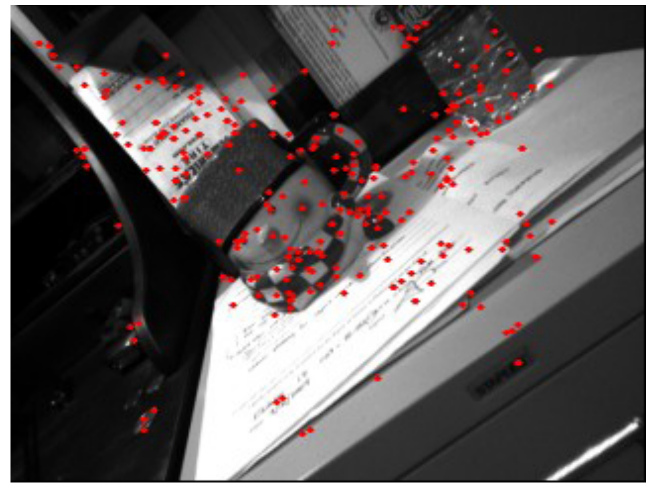


EKF Tracking Example (J. Ma Thesis)

- Can you estimate the 6DOF pose of the object in the scene?

EKF Tracking Example (J. Ma Thesis)

- Match SIFT features from model to current image



model features (*database*)

$$\mathbf{b}_j = [\underbrace{b_x, b_y, b_z}_{\text{Feature location (in object frame)}}, d_j]$$

SIFT descriptor

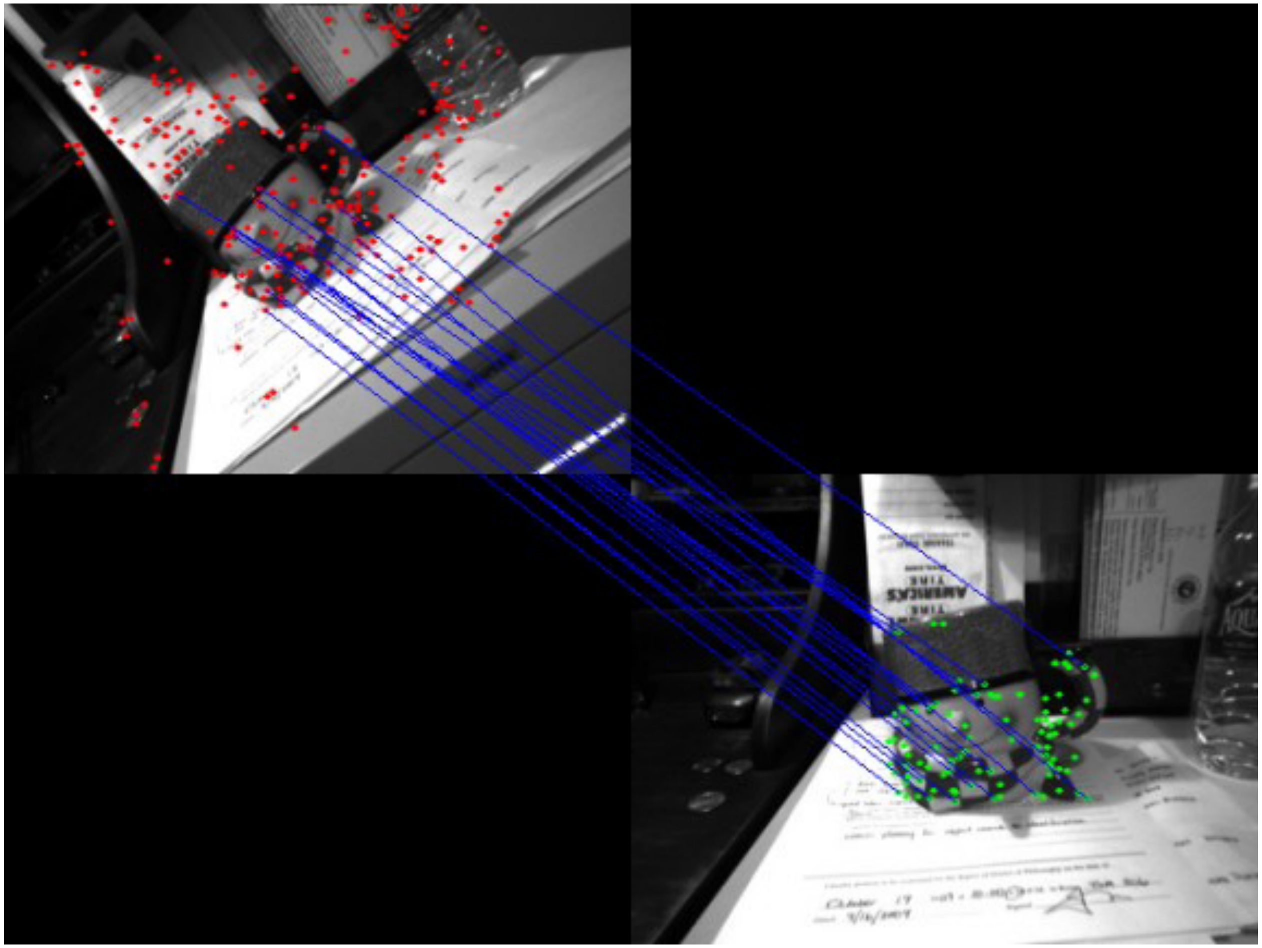
current image features

$$\bar{d}_j$$

Observed descriptor

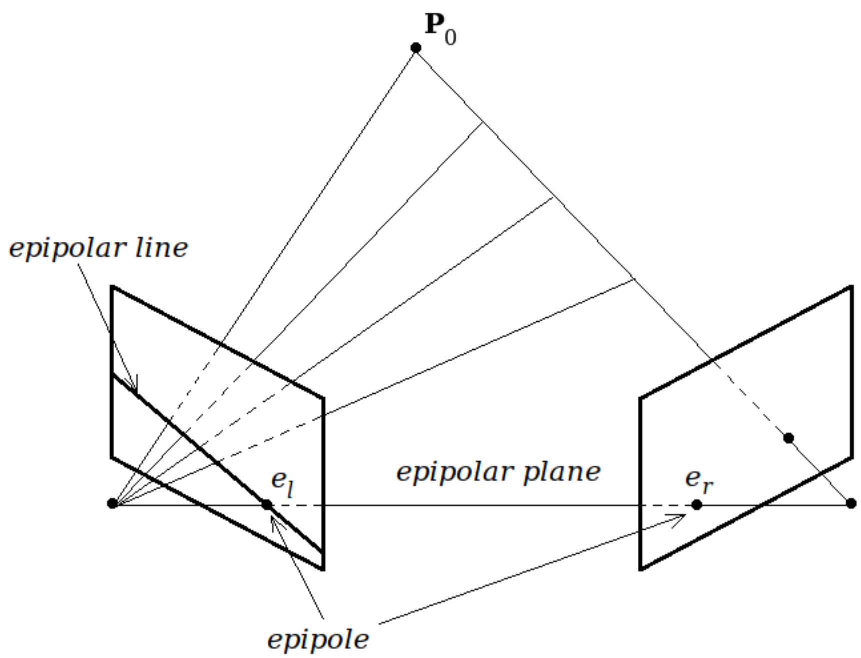
EKF Tracking Example (J. Ma Thesis)

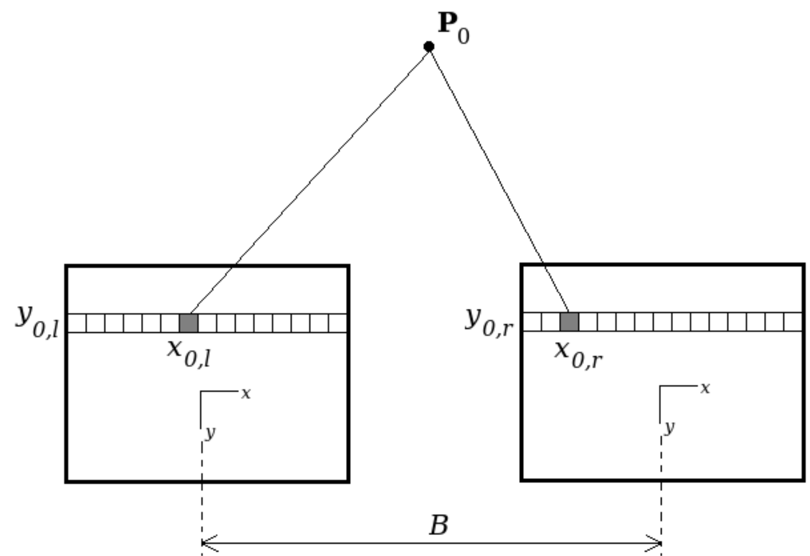
- We will assume perfect matching between observed features and the model features *Best-Bin-First Search Method*



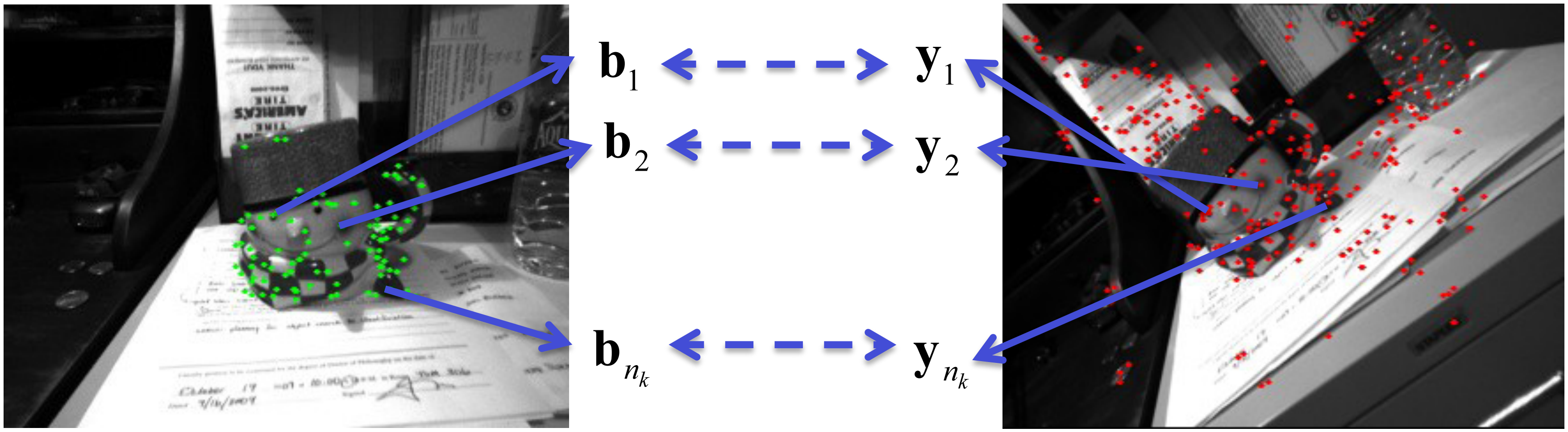
Key Idea- can measure “3D location” of points

- Using stereo cameras, the 3D location of SIFT features can be measured (in CAMERA FRAME)





Measurement Model



$$\mathbf{b}_j = [b_x, b_y, b_z]$$

$$\mathbf{y}_j = [y_x, y_y, y_z]$$

End up with a set of observed feature locations and corresponding to model features

Measurement Model

Measurement is just "object features" rotated and translated into camera frame

$$\mathbf{b}_j = [b_x, b_y, b_z]$$

$$\mathbf{y}_j = [y_x, y_y, y_z]$$

$$D_k = \begin{bmatrix} \mathbf{y}_1 \\ \mathbf{y}_2 \\ \vdots \\ \mathbf{y}_{n_k} \end{bmatrix} = \begin{bmatrix} \mathbf{x}_k + R_k \mathbf{b}_1 \\ \mathbf{x}_k + R_k \mathbf{b}_2 \\ \vdots \\ \mathbf{x}_k + R_k \mathbf{b}_{n_k} \end{bmatrix} + r_k$$

$$D_k = h(X_k) + r_k$$

$$X_k = [\mathbf{x}_k, \Theta_k]$$

$$\Theta_j = [\alpha, \beta, \gamma]$$

$$\mathbf{x}_k = [x_k, y_k, z_k]$$

$$R(\alpha, \beta, \gamma) = R_z(\alpha)\, R_y(\beta)\, R_x(\gamma) =$$

$$\begin{pmatrix} \cos\alpha\cos\beta & \cos\alpha\sin\beta\sin\gamma - \sin\alpha\cos\gamma & \cos\alpha\sin\beta\cos\gamma + \sin\alpha\sin\gamma \\ \sin\alpha\cos\beta & \sin\alpha\sin\beta\sin\gamma + \cos\alpha\cos\gamma & \sin\alpha\sin\beta\cos\gamma - \cos\alpha\sin\gamma \\ -\sin\beta & \cos\beta\sin\gamma & \cos\beta\cos\gamma \end{pmatrix}.$$

Linearization of Measurement Equation

$$\frac{\partial h_i}{\partial X_k} = \frac{\partial}{\partial X_k}\left[\mathbf{x}_k + R_k(\alpha,\beta,\gamma)\mathbf{b}_i\right] \qquad \frac{\partial \mathbf{h}_i}{\partial \mathbf{X}_k} = \begin{bmatrix} 1 & 0 & 0 & \frac{\partial h_1^i}{\partial \alpha} & \frac{\partial h_1^i}{\partial \beta} & \frac{\partial h_1^i}{\partial \gamma} \\ 0 & 1 & 0 & \frac{\partial h_2^i}{\partial \alpha} & \frac{\partial h_2^i}{\partial \beta} & \frac{\partial h_2^i}{\partial \gamma} \\ 0 & 0 & 1 & \frac{\partial h_3^i}{\partial \alpha} & \frac{\partial h_3^i}{\partial \beta} & \frac{\partial h_3^i}{\partial \gamma} \end{bmatrix}$$

$$\frac{\partial h_1^i}{\partial \alpha} = -s\alpha c\beta \cdot b_x^{J(i)} + (-s\alpha s\beta s\gamma - c\alpha c\gamma) \cdot b_y^{J(i)} + (-s\alpha s\beta c\gamma + c\alpha s\gamma) \cdot b_z^{J(i)} ,$$

$$\frac{\partial h_1^i}{\partial \beta} = -c\alpha s\beta \cdot b_x^{J(i)} + c\alpha c\beta s\gamma \cdot b_y^{J(i)} + c\alpha c\beta c\gamma \cdot b_z^{J(i)} ,$$

$$\frac{\partial h_1^i}{\partial \gamma} = (c\alpha s\beta c\gamma + s\alpha s\gamma) \cdot b_y^{J(i)} + (-c\alpha s\beta s\gamma + s\alpha c\gamma) \cdot b_z^{J(i)} ,$$

$$\frac{\partial h_2^i}{\partial \alpha} = c\alpha c\beta \cdot b_x^{J(i)} + (c\alpha s\beta s\gamma - s\alpha c\gamma) \cdot b_y^{J(i)} + (c\alpha s\beta c\gamma + s\alpha s\gamma) \cdot b_z^{J(i)} ,$$

$$\frac{\partial h_2^i}{\partial \beta} = -s\alpha s\beta \cdot b_x^{J(i)} + s\alpha c\beta s\gamma \cdot b_y^{J(i)} + s\alpha c\beta c\gamma \cdot b_z^{J(i)} ,$$

$$\frac{\partial h_2^i}{\partial \gamma} = (s\alpha s\beta c\gamma - c\alpha s\gamma) \cdot b_y^{J(i)} + (-s\alpha s\beta s\gamma - c\alpha c\gamma) \cdot b_z^{J(i)} ,$$

$$\frac{\partial h_3^i}{\partial \alpha} = 0 ,$$

$$\frac{\partial h_3^i}{\partial \beta} = -c\beta \cdot b_x^{J(i)} - s\beta s\gamma \cdot b_y^{J(i)} - s\beta c\gamma \cdot b_z^{J(i)} ,$$

$$\frac{\partial h_3^i}{\partial \gamma} = c\beta c\gamma \cdot b_y^{J(i)} - c\beta \cdot s\gamma \cdot b_z^{J(i)} .$$

Motion Model

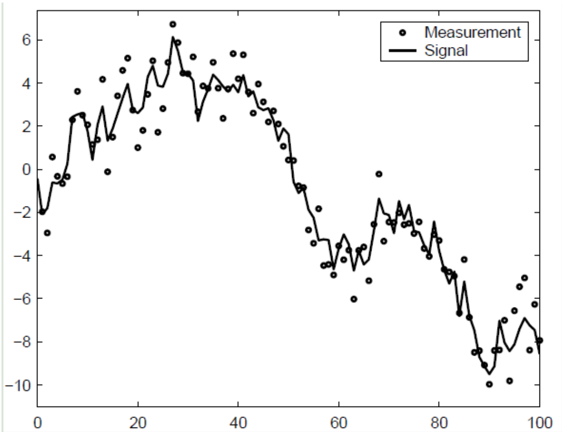
Motion model (prediction) is just a random walk
–it is already linear

$$X_k = X_{k-1} + q_k$$

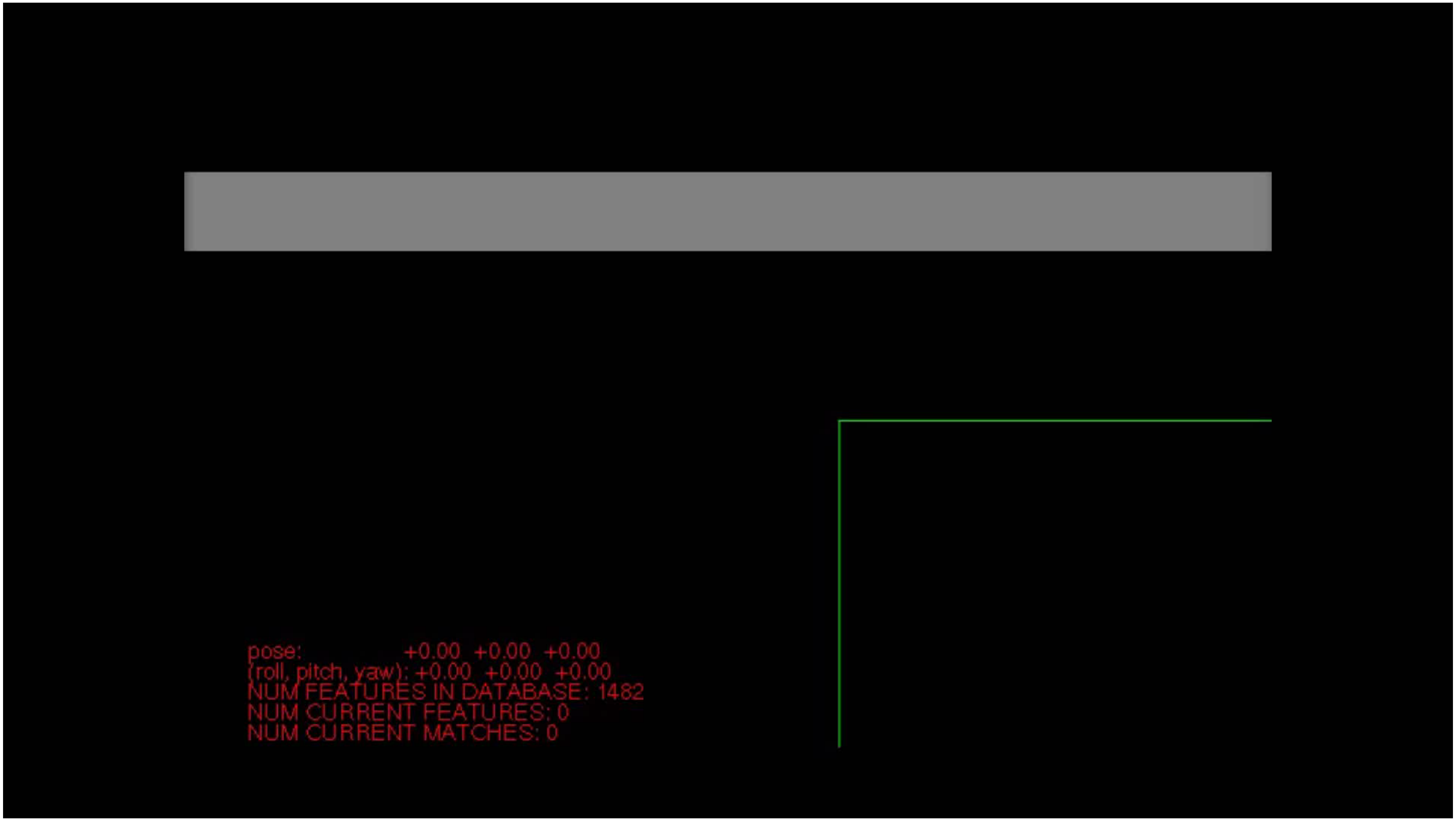
$$X_k = [\mathbf{x}_k, \Theta_k]$$

$$\Theta_j = [\alpha, \beta, \gamma]$$

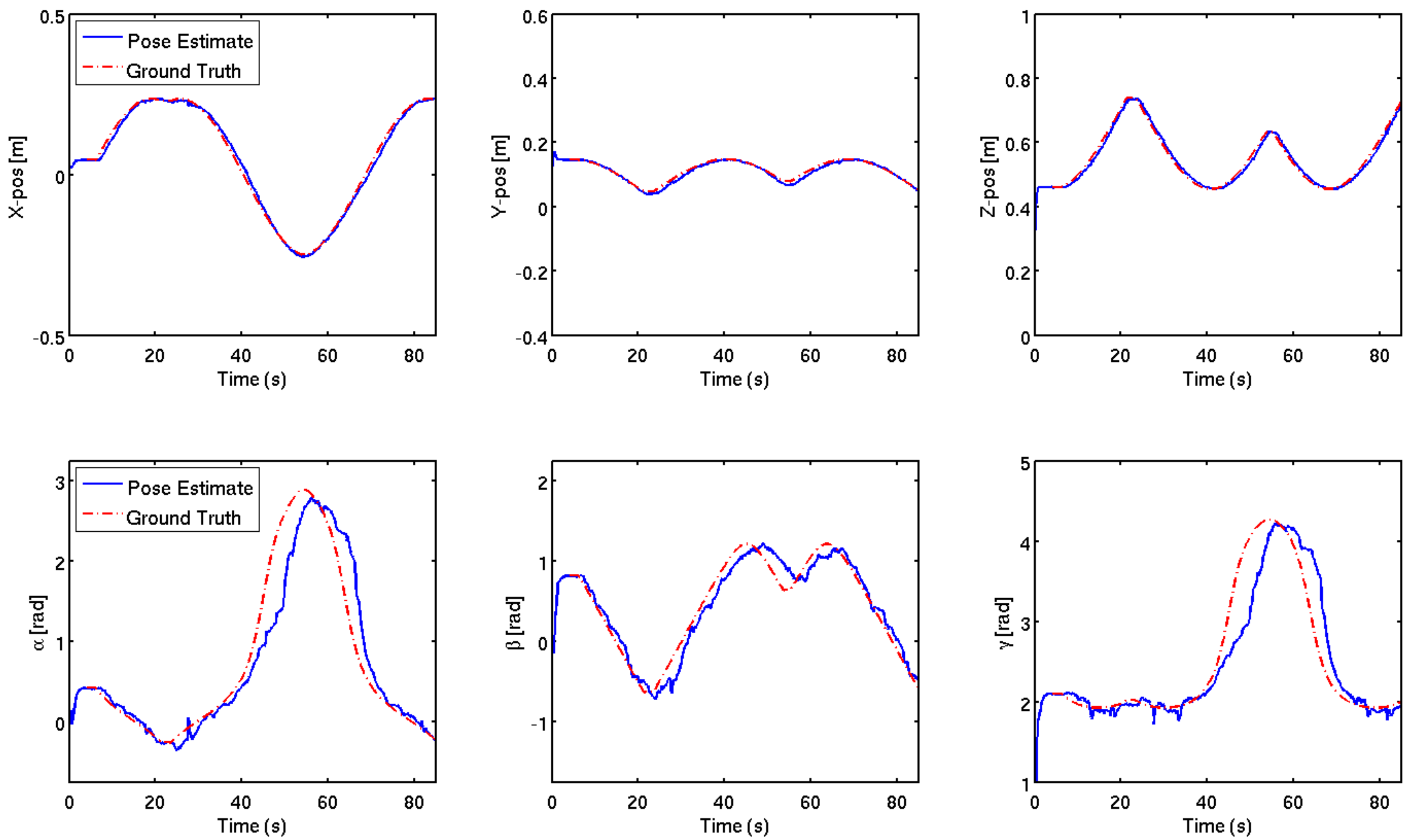
$$\mathbf{x}_k = [x_k, y_k, z_k]$$



EKF Results


pose: +0.00 +0.00 +0.00
(roll, pitch, yaw): +0.00 +0.00 +0.00
NUM FEATURES IN DATABASE: 1482
NUM CURRENT FEATURES: 0
NUM CURRENT MATCHES: 0

EKF Example


Pose Estimate
Ground Truth
X-pos [m]
Time (s)
Y-pos [m]
Time (s)
Z-pos [m]
Time (s)
Pose Estimate
Ground Truth
α [rad]
Time (s)
β [rad]
Time (s)
γ [rad]
Time (s)

Further EKF Examples:

- Brian will show you a EKF for mapping and localization
- Uses a similar concept of linearizing the rotation matrix associated with both the propagation and update equations.

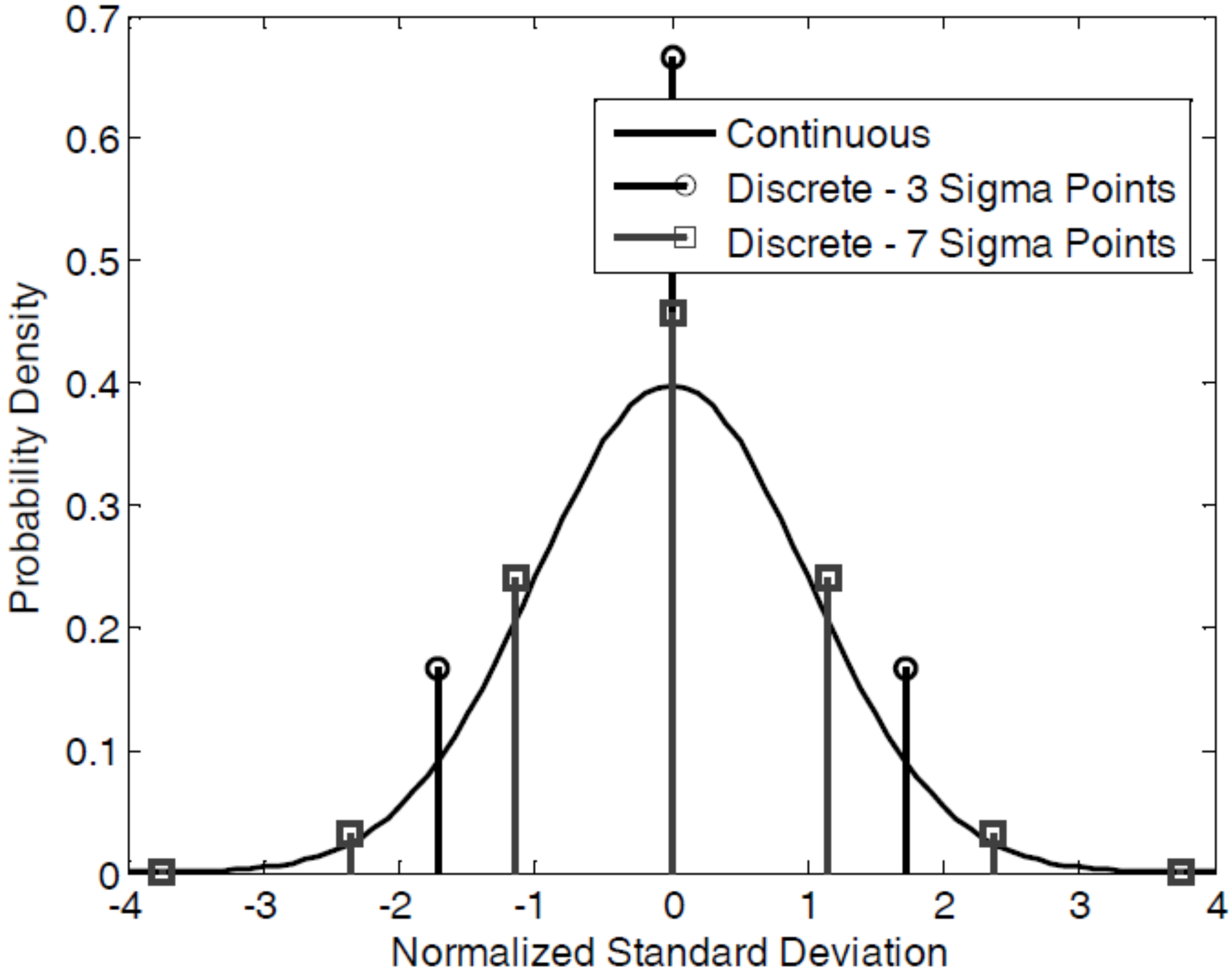
THE UKF

- Another way to linearize a system is through the Unscented Transform.
- This creates the **Unscented Kalman Filter** (or Sigma Point Kalman Filter)

- **Read SS 3.4 in "Probabilistic Robotics"**

- Is also highly efficient (not quite as good as the EKF)
- If you don't like derivatives (or you problem does not enable efficient representations of them) it is also derivative free!

Sigma Points



Sigma Points

Intuition (Julier & Uhlmann, 2004):
It is easier to approximate a probability distribution than it is to approximate an arbitrary nonlinear function.

For a 'n' dim Gaussian:

Sigma points

$$\chi^0 = \mu$$

$$\chi^i = \mu + \left(\sqrt{(n+\lambda)\Sigma}\right)_i$$

$$\chi^{i+n} = \mu - \left(\sqrt{(n+\lambda)\Sigma}\right)_i$$

Weights

$$w^0 = \frac{\lambda}{n+\lambda}$$

$$w^i = \frac{1}{2(n+\lambda)}$$

$$w^{i+n} = w^i$$

$\left(\sqrt{A}\right)_i$ i^{th} column of matrix square root

λ User defined constant

$$\mu = \sum_{i=0}^{2n} w^i \chi^i$$

$$\Sigma = \sum_{i=0}^{2n} w^i (\chi^i - \mu)(\chi^i - \mu)^T$$

Sigma Points

Intuition (Julier & Uhlmann, 2004):
It is easier to approximate a probability distribution than it is to approximate an arbitrary nonlinear function.

Pass sigma points through nonlinear function

$$\psi^i = g(\chi^i)$$

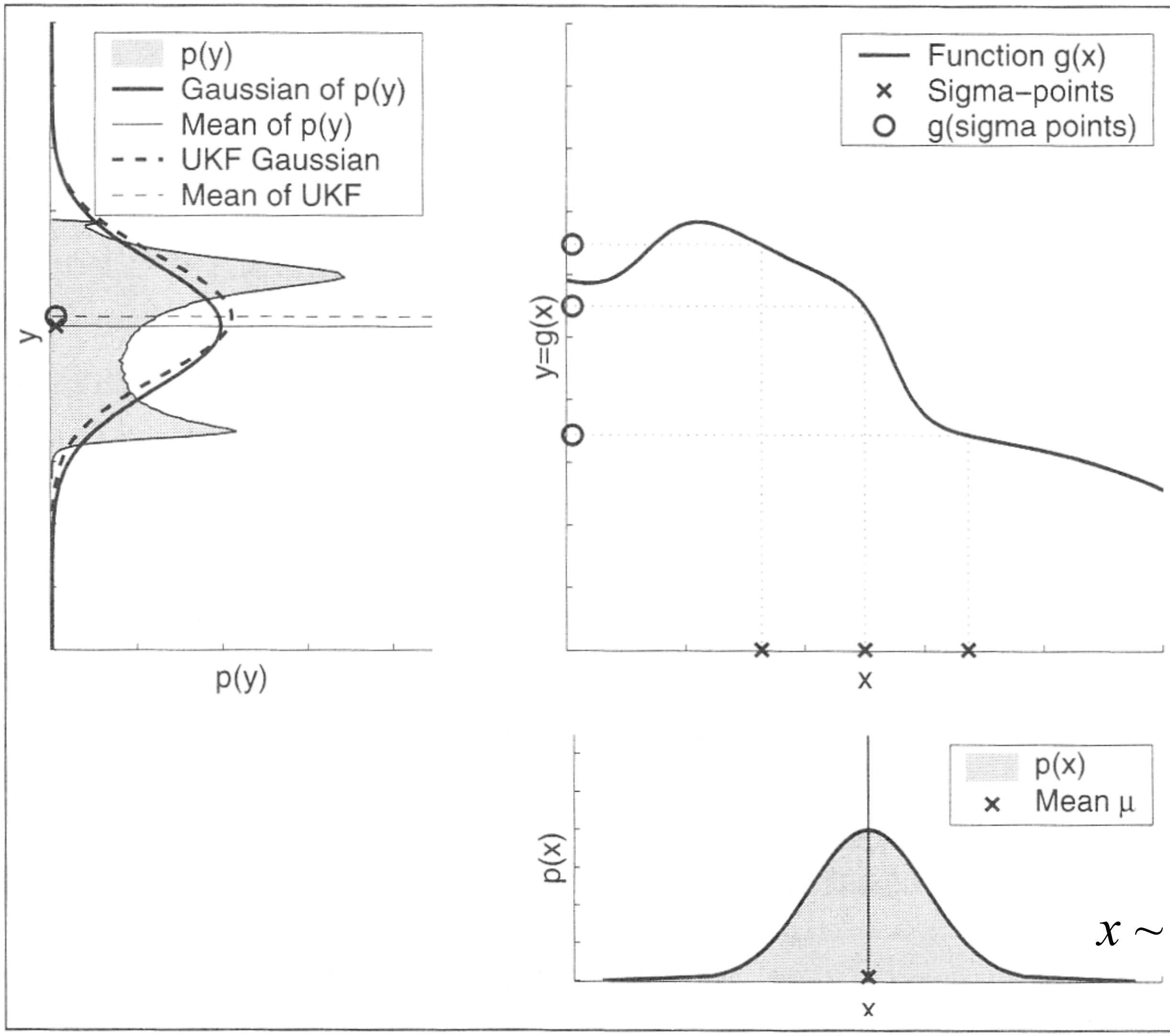
Recover mean and covariance

$$\mu' = \sum_{i=0}^{2n} w^i \psi^i$$

$$\Sigma' = \sum_{i=0}^{2n} w^i (\psi^i - \mu')(\psi^i - \mu')^T$$

Unscented Transform

$$y = g(x)$$



$$x \sim N(\mu, \sigma^2)$$

Prediction and Update Formulas

Prediction:

$$N\left(\mu_{k-1},\Sigma_{k-1}\right)$$

$$\boxed{p(x_k \mid z_{1:k-1}) = \int p(x_k \mid x_{k-1})\, p(x_{k-1} \mid z_{1:k-1})\, dx_{k-1}}$$

$$\chi_{t-1} = \left[\mu_{t-1} \quad \mu_{t-1} + \gamma\sqrt{\Sigma_{t-1}} \quad \mu_{t-1} - \gamma\sqrt{\Sigma_{t-1}}\right] \qquad \gamma = \sqrt{n+\lambda}$$

$$\chi_k^* = g(\chi_{k-1}, u_k)$$

Generate & propagate sigma points

$$\bar{\mu}_k = \sum_{i=0}^{2n} w^i \ \chi_{i,k}^*$$

Replace integral with Summation

$$\bar{\Sigma}_k = \sum_{i=0}^{2n} w^i \left(\chi_{i,k}^* - \bar{\mu}_k\right)\left(\chi_{i,k}^* - \bar{\mu}_k\right)^T$$

Prediction and Update Formulas

$$\boxed{p(x_k \mid z_{1:k}) = \frac{1}{\eta} \underbrace{p(z_k \mid x_k)\, p(x_k \mid z_{1:k-1})}}$$

$$N\left(\bar{\mu}_k, \bar{\Sigma}_k\right)$$

$$\bar{\chi}_k = \left[\bar{\mu}_k \quad \bar{\mu}_k + \gamma\sqrt{\bar{\Sigma}_k} \quad \bar{\mu}_k - \gamma\sqrt{\bar{\Sigma}_k}\right] \qquad \gamma = \sqrt{n+\lambda}$$

$$\bar{Z}_k = h(\chi_{k-1})$$

Generate & propagate sigma points

$$\hat{z}_k = \sum_{i=0}^{2n} w^i \, \bar{Z}_{i,k}$$

Predicted measurement mean

$$S_k = \sum_{i=0}^{2n} w^i \left(\bar{Z}_{i,k} - \hat{z}_k\right)\left(\bar{Z}_{i,k} - \hat{z}_k\right)^T$$

Pred. measurement covariance

Prediction and Update Formulas

$$\hat{z}_k = \sum_{i=0}^{2n} w^i \, \bar{Z}_{i,k}$$ Predicted measurement mean

$$S_k = \sum_{i=0}^{2n} w^i \left(\bar{Z}_{i,k} - \hat{z}_k\right)\left(\bar{Z}_{i,k} - \hat{z}_k\right)^T$$ Pred. measurement covariance

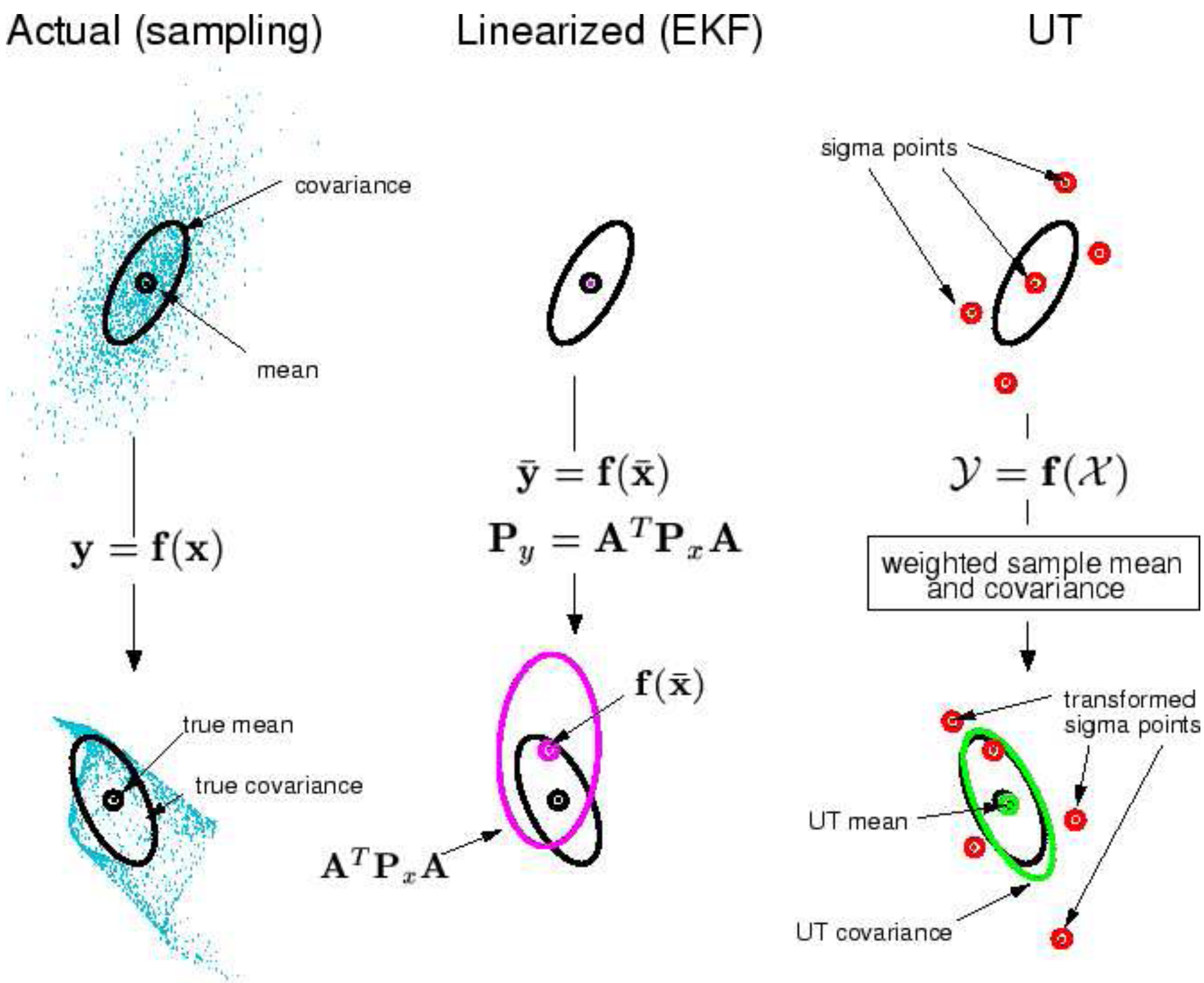
$$\Sigma_k^{x,z} = \sum_{i=0}^{2n} w^i \left(\bar{\chi}_{i,k} - \bar{\mu}_t\right)\left(\bar{Z}_{i,k} - \hat{z}_k\right)^T$$ Cross-covariance

$$K_k = \Sigma_k^{x,z} S_k^{-1}$$ Kalman gain

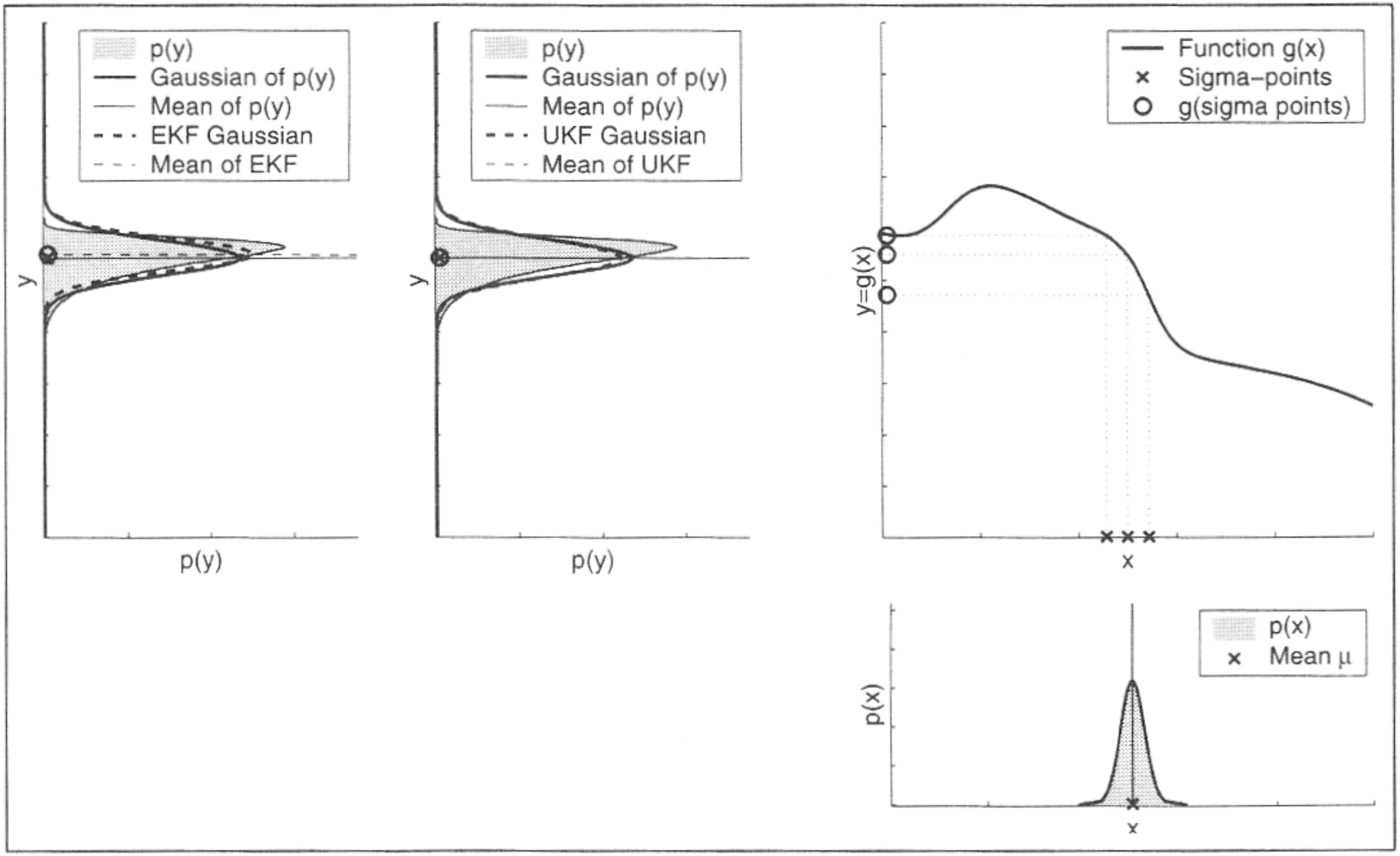
$$\mu_k = \bar{\mu}_k + K_k(z_k - \hat{z}_k)$$ Updated mean

$$\Sigma_t = \bar{\Sigma}_t - K_t S_t K_t^T$$ Updated covariance

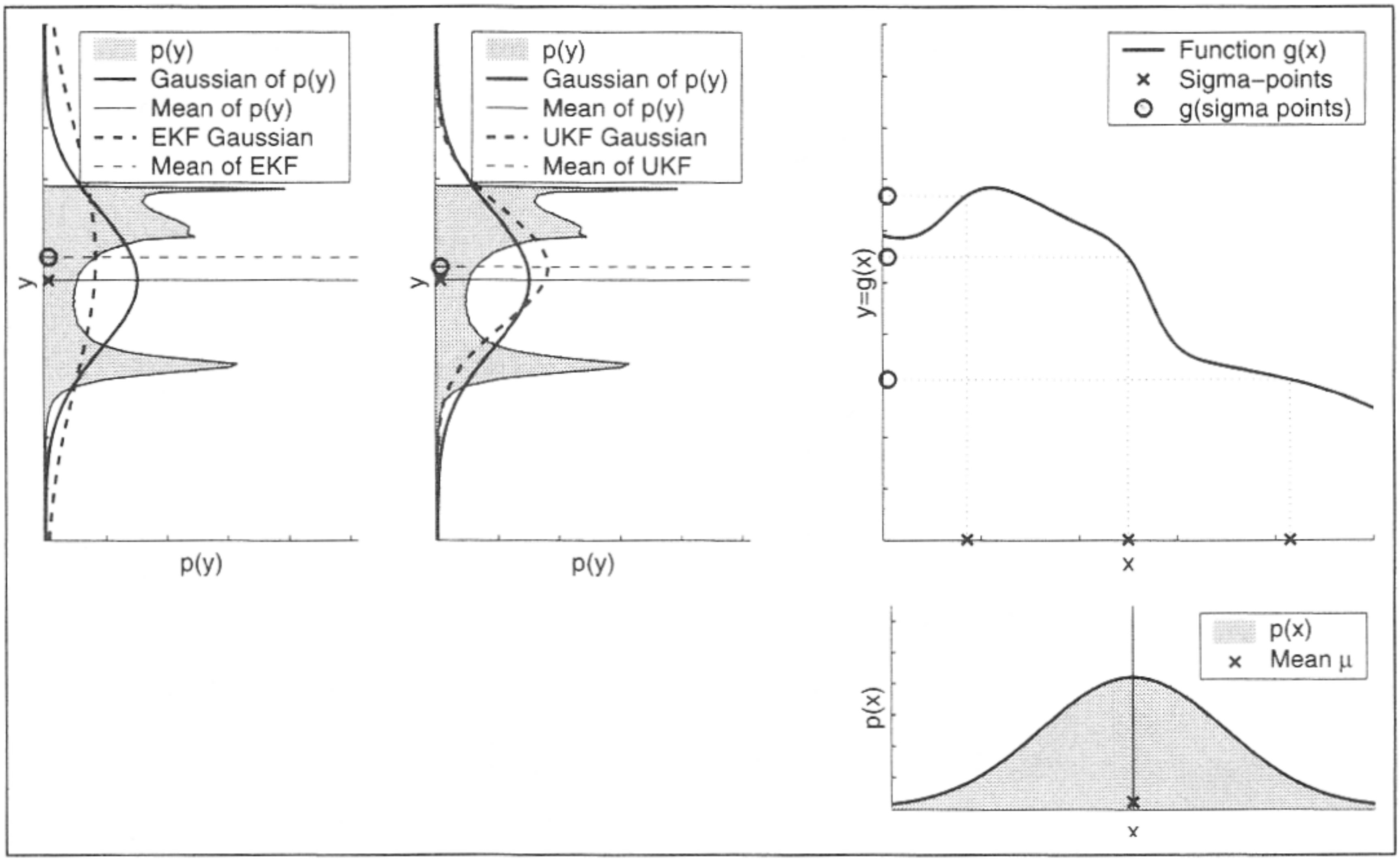
EKF, UT


Actual (sampling)
Linearized (EKF)
UT
covariance
mean
$\mathbf{y} = \mathbf{f}(\mathbf{x})$
true mean
true covariance
$\bar{\mathbf{y}} = \mathbf{f}(\bar{\mathbf{x}})$
$\mathbf{P}_y = \mathbf{A}^T\mathbf{P}_x\mathbf{A}$
$\mathbf{f}(\bar{\mathbf{x}})$
$\mathbf{A}^T\mathbf{P}_x\mathbf{A}$
sigma points
$\mathcal{Y} = \mathbf{f}(\mathcal{X})$
weighted sample mean and covariance
transformed sigma points
UT mean
UT covariance

Linearization vs. UT (1)



Linearization vs. UT (2)



Why use a UKF

- Use a UKF when
  - g &h are non-linear
  - A EKF is not performing well or you cannot derive it.

- Use a KF if linear
- Use a EKF if you can (faster – although not by much)
- Use a Particle Filter (Thursday) when the Gaussian assumption breaks down.